UTILIZING MULTIPLE ENTITIES FROM COLLECTION OF UNSTRUCTURED DOCUMENTS IN CONSTRUCTING ATTRIBUTE-VALUE PAIRS

BY

HYUN DUK CHO

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2012

Urbana, Illinois

Adviser:

Associate Professor Chengxiang Zhai

ABSTRACT

Attribute-value pairs, or NVP is defined as extracting words expressing characteristics of entity and associating the said words with words or phrases that best describe the attributes. Applications for NVP arise in various related area such as sentiment analysis, populating and checking for errors in relational database to a broader text information area such as QA systems, search and review modeling.

We propose an unsupervised method to identify the properties of entities represented as NVP from unstructured documents. Other approaches that extract NVP usually utilize supervised or semi-supervised approaches on structured or semi-structured documents. Benefits of such approaches lie in that they tend to have higher accuracy than unsupervised approaches on unstructured documents. Furthermore, supervised approaches are more suited to distinguishing attribute words to that of value words than unsupervised approaches on unstructured documents. The biggest drawback with the said methods however, is that training data may not always be available and not all documents can be thought of as being unstructured.

We first proposes in this thesis an approach to extracting and distinguishing attribute words and value words from unstructured documents. Since entities of the same class share similar attributes, we propose that the identification of relevant attributes should be done across entities belonging to the same class, and demonstrate that this can lead to a significant performance gain in attribute extraction, even when only documents describing a modest number of entities per class is available.

We then propose a way to evaluate the accuracy of attribute-value pairs automatically, allowing for quantitative comparison between different systems that is more consistent and cost-effective than manual evaluations. These were used in evaluating summarization or

comparing ontologies. However, these techniques have not been utilized in evaluating NVP. Both the automated and manual evaluations show that our system outperforms a comparison system.

To my parents, for their love and support.

ACKNOWLEDGMENTS

It gives me great pleasure in acknowledging the support and help of Professor Chengxiang Zhai, without whom this work would not have been possible. His comments, patience, knowledge, and attention to detail motivates and encourages an academic atmosphere.

I'd also like to thank Professor Julia Hockenmaier. Her insights from natural language processing perspective was invaluable in writing up this thesis.

I cannot find words to express my gratitude to Jump Trading for its very generous scholarship/fellowship. This was overall a very beneficial program, where they not only provided two years of tuition and stipend, but also invaluable experience in cutting edge technology in the industry. I would especially like to thank Stephen Yi, who is the vice president of the company, for his initial push for the establishment of Jump Trading Scholars.

Many of my colleagues and friends have helped shaped my research. I owe greatly to Majid Kazemian, who is both a great friend and a colleague, for providing a very helpful guidance to being a successful graduate student. I would like to thank Alexander Kravchenko and Yi Cheng for the weekly Sunday breakfast club, Fred Douglas, Jing Zou and Xiaolong Wang for various activities and empathy we have had as new graduate students, and Matthew Morse for discussions that have provided insights into the thesis topic. I would like to further acknowledge Matthew Frank, Eric Murdock and Chandler Womack for their support in various non-academic matters.

Finally, I cannot express in words how much support I take for granted from my parents. My father have shown me courage to tackle on problems no matter how grave the situation may look like, and my mother for her countless number of words of encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

When users seek information about entities (e.g. products they may wish to purchase), they are often interested in their attributes or properties (e.g. the technical specifications of a laptop), and wish to compare different entities of the same class (e.g. different models of laptops) according to those attributes. In order to facilitate such comparisons, we wish to automatically extract relevant attributes (such as CPU, memory) and their values (2.6 Ghz, 4GB) from raw text, allowing the automatic generation of attribute-value pairs, or NVP tables such as Figure 1.1. For ease of evaluation, we focus here on entity classes whose properties are known (such as laptops, whose technical specifications are well documents), but note that other kinds of entities, such as hotels, resumes, or drugs, whose positive and negative effects may be discussed by patients [1] can especially benefit from having an algorithm to extract attribute-value pairs.

The attribute-value pairs we extract can also be used in question answering (QA) systems [2], for query expansions, and to populate relational databases. Furthermore, systems that extract NVP can be used in checking for errors made by existing systems. Often times online retail shops such as Amazon or Newegg receive unstructured product specification catalogs from manufacturers. These are then entered into the internal database manually. Human errors may be introduced during the process. It is possible for customers to become disgruntled from these errors, potentially becoming a disastrous event for retail shops. An error correction system by leveraging existing NVP and newly obtained NVP can help mitigate such events.

Model#	1015PX-SU17-WT	1016PT-BU37-BK
Brand	ASUS	ASUS
Series	Eee PC	Eee PC
Color	White	Black
OS Provided	Windows 7 Home Premium 32-bit	Windows 7 Professional 32-bit
CPU Type	Intel Atom	Intel Atom
CPU Speed	N570(1.66GHz)	N455(1.66GHz)
CPU L2 Cache	1MB	512KB
Chipset	Intel NM10	Intel NM10
Screen Size	10.1"	10.1"

Figure 1.1: An example of a desirable output.

1.2 Challenges

There are two major challenges in extracting attribute-value pairs. First, almost all documents on the web consist of largely unstructured text [3], and we cannot assume structured documents (i.e. tables) for particular class of instances to exist. Even if such documents may exist, learning which part of the table specifies attributes and which one specifies their values is a challenge by itself [4, 5, 6].

Identifying attributes and values from raw text is challenging, since web documents (especially user-generated reviews, which are of particular interest for this task) tend to contain choppy, and often grammatically incorrect sentences. Approaches based on lexico-syntactic pattern matching [7, 8] may need further improvements to have high recall and reasonable precision at web scale [4], whereas sophisticated NLP techniques may not always be feasible or accurate [9].

Second, due to the large number of potential entity classes one may wish to compare (and the fact that attributes can be very specific to the classes of interest), we cannot rely on supervised approaches. Unsupervised approaches are advantageous for our purposes because they do not require labeled in-domain training data for every entity class of interest.

1.3 Thesis Contributions

In this thesis, we address the two challenges discussed above. Specifically we provide solutions to circumvent the lack of lexical patterns in web documents and the low accuracy of unsupervised attribute-value extraction systems. In order to improve the performance of unsupervised attribute-value extraction systems, we propose to take advantage of the redundancy of attributes across multiple entities of the same class. Leveraging redundancy in a supervised system for structured data has been investigated in a related work [4], but we demonstrate its usefulness on the much harder task of unsupervised extraction of attribute-value pairs from unstructured text. Furthermore, we improve upon an existing work to better fit the characteristics of attributes. Red Opal [11] is based on word statistics based on Poisson distribution. We note how attribute words are likely to be distributed evenly across the documents and further improve upon their work using the observation. First, we add a term to determine how frequently words appear across documents to penalize those that appear in only small set of documents. Next we introduce global and local attribute extractors which we show helps in balancing recall and precision for a better F-score.

We further note that it is beneficial to have an automated approach in evaluating attribute-value pairs. Relying on humans to evaluate system performance is costly, and not easily done on a frequent basis [10]. In addition, since some papers use crowd sourcing [11] while others employ professionals [12], a direct comparison between the results reported in different papers is not possible. We thus propose a novel metric that leverages existing gold standards to evaluate the extracted attributes, values, and attribute-value pairs. To the best of our knowledge, there do not exist such automated evaluation metric in evaluating all three of these at once.

The thesis is organized as follows. We discuss background and related works in Chapter 2. In Chapter 3, we define the terminology used throughout the thesis, and describe our algorithm in depth. Evaluation is described in detail in Chapter 4. We note in this section that automatic evaluation can be done by utilizing an existing recall-based evaluation technique [10], as opposed to human evaluation, which may lack standardization. Finally, we conclude and describe future works in Chapter 5.

CHAPTER 2

BACKGROUND

Attribute-value pairs have various applications in both traditional and non-traditional information retrieval, e.g. for product reviews [13, 14], sentiment analysis [15], summarization [16] and question answering [2]. They have also been shown to be useful for other tasks, e.g. when determining the effects of drugs [1], or obtaining contrastive opinions [17, 18] (here, attributes correspond to produce aspects or viewpoints, and values are defined as lists of contrastive opinions for particular attributes).

In order to extract attribute-value pairs, it is often the case that attributes of interest are identified first, and then values associated with the mined attributes are extracted. While attributes of entities can be classified into various roles [19], in this thesis, we are mainly interested in 'constitutive' attributes, which include physical properties (e.g 'engine' and 'battery' for automobiles).

There are numerous works done in attribute extractions. Early approaches [8, 20] rely on pattern matching, and tend to perform poorly on web documents. Some approaches [21, 22, 23] rely on Latent Dirichlet Allocation (LDA) [24], while others exploit the implicit structure of HTML documents [25, 26], or structured data such as tables [4, 27, 28] to aid attribute extractions. While such approaches often outperform methods that do not assume any underlying structure, they may not be applicable to all domains of interest. Furthermore, not all HTML documents provide sufficient structure, and these approaches may fail in such cases. Statistical models [11, 20] and pattern mining approaches [13] use the relative frequencies of nouns to determine attribute terms. Such model tends to be light, fast, and provide a language independent approach to extracting attributes. We use statistical approach shown in [11] to extract attribute words.

There exists some work done in attribute-value extractions. Many approaches try to utilize

structured [25, 4, 27] or semi-structured data [29, 7], and are often supervised or semi-supervised. One line of work assumes that structured product descriptions, or pre-existing seed values, are available to extract attribute-value pairs from [7, 26]. Other approaches leverage Wikipedia [29], and have shown that Wikipedia infoboxes are a reliable source for the identification of attribute-value pairs. However, all of these approaches fail when the required information (e.g. structured seed data or Wikipedia infoboxes) is not available.

Similar to the work presented here, some approaches aim to use unsupervised methods [12, 8] which make minimal assumptions about the documents and domains of interest, and therefore promise better coverage than other techniques. Pattern based approaches [8, 12] work well in cases where documents have well-defined grammatical structures, but suffer from low recall on text that is less well edited, including most user-generated content on the web. A web query approach [12] attempts to work around this limitation by utilizing the entire web, as opposed to selected subset of documents to build attribute-value pairs, and performs well on well defined topics, such as digital cameras. Yet, it may fail on more specific topics (such as specific models of digital cameras), because it becomes harder to match patterns as topics become more specific and documents matching these patterns become more sparse.

CHAPTER 3

ATTRIBUTE AND VALUE EXTRACTION

In this chapter, we formally define attribute and value extraction problem, and propose an unsupervised approach by utilizing redundancy. For any given class of entities c, let us denote $t_i \in subClasses(c)$, where all the entities of interest, t_i are contained within the subclasses of a class c. Attributes of entity t_i are denoted as $a_{ij} \in A_i = attributes(t_i)$. The attributes A_g of the superclass c are given by $a_{gj} \in A_g = A_1 \wedge A_2 \wedge \cdots \wedge A_n$ for all the subclasses in c. The values associated with attribute a_{ij} are defined as $v_{ijk} \in values(a_{ij})$. We note here that we allow entities to have multiple values for the same attribute, allowing for more flexibility of what values the attributes may have, which we provide an example of later in this section.

The problem we are attempting to solve then, is to extract tuples (a_{gj}, v_{ijk}) $\forall i, j, k$ from a set of documents D. We assume each document describes a single (known) entity, and define D_i as the set of documents that is related to entity t_i, and assume $D_i \cap D_j = \phi$, $i \neq j$. We note that we know which documents are related to which entity beforehand as determining different entities in set of documents can be automated [30] or preprocessed beforehand, and hence is not a focus in this thesis.

As an illustrative example, let us assume we have a list of models of laptops. Each different model is denoted as t_i, and the class 'laptop' is denoted as c. Each model has a number of attributes (such as CPU or RAM) that all models possess (albeit with possibly different values), whereas other attributes (such as the resolution of the webcam) only apply to a subset of models, and are hence not included among the attributes for the parent class c. Finally, some of the attributes, such as 'CPU', may take multiple values for the same entity, since they can refer to the raw speed of the processor (e.g. 1.6GHz), or a specific CPU model (e.g. Intel Core i5), and in the absence of more specific attribute labels, we would like to

extract both values for the same attribute.

3.1 Attribute Extraction

Our approach to attribute extraction exploits the fact that attributes are shared amongst entities of the same class. Using this intuition, we introduce two types of attribute extractors. The first extracts attributes using all the documents in the collection, while the second type extracts them from all documents D_i associated with entity t_i. We then combine both extractors to retrieve those attributes that best fit the entity. This is a natural extension of leveraging redundancy of attributes while avoiding potential noise words detected by the first type of attribute extractor.

3.1.1 Core Attribute Extraction Algorithm

Our work extends the statistical attribute extraction method called Red Opal [11]. Red Opal is a system which scores different products of the same class based on the attributes that the product has, and ranks them according to each of the retrieved attributes. For example, if a user is interested in smartphones, the attributes of interest may be battery life, operating systems or price. The system then retrieves products that have high scores for the chosen attribute. Red Opal postulates that words are generated independently according to a Poisson distribution, and that if any word appears more often than can be expected by chance, it is likely that this word describes some attribute.

They note that for large corpus size N and independently generated words x with relative frequencies $p_x = \frac{n_x}{N}$ (where n_x is the frequency of word x), the Poisson distribution reduces to a binomial distribution, whose natural logarithm can be approximated by Stirling's approximation to yield Equation 3.1:

$$\ln\ P(n_x) \approx (n_x - p_x N) - n_x \ln(\frac{n_x}{p_x N}) - \frac{\ln(n_x)}{2} \tag{3.1}$$

The authors note that if $n_x > p_x N$ and $\ln(P(n_x))$ is small, it is unlikely x occurred so

often by chance, and hence categorize the word as an attribute word. Finally, the authors note that product features are generally nouns and they thus use only the nouns to extract potential attribute words. Statistical approaches such as Red Opal may suffer from a lack of means to distinguish between attribute words and value words because they are based on relative word frequency, and adding redundancy across entities will ensure these relative word frequency is especially high for attribute words. It should be noted however that such technique is general, and can be applied to other attribute extraction approaches.

3.1.2 Analysis and extensions of Red Opal

Note that Red Opal does not take the number of documents in which a word appears into account, and may therefore select words that appear with a very high frequency in a small number of documents. However, since attributes are shared by many entities, we postulate that they tend to appear in many different documents. As an illustrative example, consider a list of laptops from various companies. It is easy to see that attribute words tend to appear often across documents, while value words occur only in the subsets of documents that pertain to specific entities. For instance, the word 'Apple' may appear with high frequency in the subset of documents that relate to MacBooks, even though it does not denote an attribute, but a value (of the attribute 'manufacturer'). However, 'Apple' do not appear often in other company's laptop documents. Red Opal however, do not consider this cross document word appearance into account, and some value words may actually appear often enough that it may be considered an attribute word.

We therefore propose to improve upon Red Opal by introducing a way to take words appearing across documents into account. We calculate entropy for document given candidate word w. High entropy implies that a particular word appears almost evenly across the different documents while low one implies only small subset of documents have the said word. Attribute words are likely to be evenly distributed amongst the document of interest, and hence we propose adding entropy term.

Before calculating the entropy, we make a simplifying, yet intuitive assumption that the probability $p(d)$ is uniform where $d \in D$ is document. While not explicitly used in the

formulas, this ensures that the weights given to each documents contributes exactly the same to the cross document entropy measure. Our scoring function is based on conditional entropy $H(D \mid w)$, and it is formulated as follows:

$$H(D \mid w) = -\sum_{d} p(d \mid w) log\ p(d \mid w) \tag{3.2}$$

Finally, we combine Equation 3.1 and Equation 3.2 into a new attribute scoring function, shown by Equation 3.3

$$score(w) = lnP(n_w) * H(D \mid w) \tag{3.3}$$

The effects of cross document entropy term are shown in Section 4.

3.1.3 Attribute Extraction

We extract attributes by combining global and local attributes. Using only local attribute extractor to obtain attributes may lead to the algorithm extracting both attributes and values. It should be noted that unsupervised attribute extractors do not have explicit approach of distinguishing between attribute and value, leading to disambiguities in choosing what is attribute and what is value. Using only the global attribute extractor on the other hand may lead to potentially detecting noise words as being important. There may exist a word that was not ranked high enough in local attribute extractors that may rank high enough globally, leading the algorithm to believe such word may have been an attribute word.

Global attributes are extracted by using Equation 3.3 from entire set of documents. We call attributes extracted from this as A_g. For each entities t_i, we extract attributes, again using the same equation. We call these attributes A_i. We now provide a pseudo-code detailing our attribute extraction in Algorithm 1.

We note that if we know the list of global attributes before hand, then Equation 3.4 holds.

$$(A_g \cap A_i) \cup (A_g \cap A_j) \subseteq A_g\ \forall i, j,\ i \neq j \tag{3.4}$$

Algorithm 1 Attribute Extraction

Input: Documents $d_{il} \in D \; \forall i, l$
Output: List of class-wide attributes A
1: $A = \phi$
2: A_g = attributes from Equation 3.3 by using D
3: **for all** D_i from D **do**
4: $\quad A_i$ = attributes from Equation 3.3 by using D_i
5: $\quad A = A \cup (A_g \cap A_i)$
6: **end for**

Line 5 of Algorithm 1 takes the above equation into account, and placed them into list of attributes that we are interested in. Taking intersections of attributes with each of A_i allows removing potential attributes from A_g that may have been a noise if it did not appear in A_i.

It can be argued that instead of using Equation 3.4, a mere intersection of A_i, or using A_g as list of attributes may have sufficed. We show experimentally in Section 4 that combining global and local attribute extractor is the best in balancing recall and precision In the section, we show that taking intersection of A_i results in low recall but high precision, and using only A_g results in high recall but low precision.

3.2 Value Association

Value association is defined as assigning words or phrases that are related to attribute of interest. It is not surprising to note that values associated with attributes are short, since users are most likely looking for bird-eye's view of the particular aspect and explanations that are too long would not be very interesting to users. Furthermore, values are likely to be replicated across documents. It is unlikely that values of attribute appear in only a very small subset of documents explaining entity t_i. If that were the case, such may not have been an interesting value in the first place. Finally, it is unlikely that attribute words and value words are separated far apart when these are being described in unstructured documents. Based on the above observations, values on unstructured documents should be

1) phrases or very short sentences

2) replicated across documents describing the same entity

3) close proximity to attribute

Our initial attempt in extracting values used pattern-based approach [8]. These approaches exploit characteristics of grammar rules that are often seen in free texts. For the pattern based approach, we note that both the rules that appear in [8] and in [12] utilize prepositions and auxiliary verbs in determining the patterns. Furthermore, the value words tend to appear after prepositions or auxiliary verbs rather than before them. As an example, '[A] of the digital camera is [V],' we see that attribute word appears before the word 'of' and value appears after the said preposition.

We compiled a list of prepositions and auxiliary verbs (is, are, was, in, at, from, above, on, has, had, have, as, by) and tested the hypothesis empirically on Wikipedia. The topic we have chosen is on nations, and attributes we show are on ocean, economy and location, which were some of the words chosen by the attribute selection algorithm. Our empirical results are shown in Figure 3.1.

It can be seen that on articles that are well structured the pattern based approach tends to perform well. However, performance of the algorithm degraded significantly on web documents, reaffirming previous findings that pattern-based approaches are not suited for extracting information from the web [31]. The biggest disadvantage to pattern-based approach is that they are inflexible to changes in semantics and small grammatic change may hurt its performance greatly. Pure statistical or word frequency based approaches on the other hand enable value phrases to be placed anywhere in documents and they are less vulnerable to grammatical mistakes of documents. However these do not take advantage of structural cues of language, potentially omitting important value words that may have been brought up. We thus propose taking advantage of structures of language while leveraging document and word frequencies that may be associated with the value of the attribute word.

In particular, our next attempt utilized text summarization technique. Text summarization [16] summarizes texts from either a single document, or collection of documents. Generated summaries should preserve important information and be short [32]. We note that such characteristics align with the first two observations of values in documents. Finally values tend to be in close proximity to attribute words as was noted in our third observation. This allows running summarization techniques only on the phrases that are close in proximity to

mexico	<on$_1$[on the southeast by guatemala , belize , and the caribbean sea ; and on the east by the gulf of mexico]>
japan	<to$_1$[to the east of the sea of japan , china , north korea , south korea and russia , stretching from the sea of okhotsk in the north to the east china sea and taiwan in the south]>
china	<is$_1$[is 14,500 kilometres -lrb- 9,000 mi -rrb- long -lrb- the 11th-longest in the world -rrb- , and is bounded by the bohai , yellow , east and south china seas]>
canada	<in$_1$[in the east to the pacific ocean in the west , and northward into the arctic ocean]>
canada	<in$_1$[in the west , and northward into the arctic ocean]>
brazil	<on$_1$[on the east , brazil has a coastline of 7,491 km -lrb- 4,655 mi -rrb-]>

Feature Word : Ocean

mexico	<is$_1$[is strongly linked to those of its north american free trade agreement -lrb- nafta -rrb- partners , especially the united states]>
japan	<by$_1$[by nominal gdp -lrb- 12 -rrb- and fourth largest economy by purchasing power parity]>
japan	<by$_1$[by purchasing power parity]>
germany	<by$_1$[by nominal gdp and the fifth largest by purchasing power parity]>
uk	<by$_1$[by nominal gdp and seventh-largest economy by purchasing power parity]>
uk	<by$_1$[by purchasing power parity]>
france	<by$_1$[by nominal gdp]>
australia	<has$_1$[has the world 's sixth-highest per capita income]>
usa	<is$_1$[is the world 's largest national economy , with an estimated 2010 gdp of $ 14.53 trillion -lrb- 23 % of nominal global gdp and over 19 % of global gdp at purchasing-power parity -rrb-]>
usa	<was$_1$[was the world 's largest]>
canada	<is$_1$[is reliant upon its abundant natural resources and upon trade -- particularly with the united states , with which canada has had a long and complex relationship]>
spain	<in$_1$[in the world by nominal gdp , and very high living standards , including the tenth-highest quality of life index rating in the world , as of 2005]>
brazil	<by$_1$[by nominal gdp -lrb- 15 -rrb- and the eighth largest by purchasing power parity]>

Feature Word : Economy

japan	<in$_1$[in the pacific ocean , it lies to the east of the sea of japan , china , north korea , south korea and russia , stretching from the sea of okhotsk in the north to the east china sea and taiwan in the south]>
china	<in$_1$[in east asia , the country covers approximately 9.6 million square kilometres -lrb- 3.7 million square miles -rrb-]>
china	<in$_1$[in the turpan depression]>
france	<on$_1$[on other continents and in the indian , pacific , and atlantic oceans]>
greece	<in$_1$[in greece , ranking greece 7th in europe and 13th in the world]>
canada	<in$_1$[in the northern part of the continent , it extends from the atlantic ocean in the east to the pacific ocean in the west , and northward into the arctic ocean]>
spain	<in$_1$[in southwestern europe on the iberian peninsula]>

Feature Word : located

Figure 3.1: Topic is on countries on attribute (feature) words ocean, economy and located. It can be seen that value phrases extracted from the article are reasonably well aligned to the attribute word.

attribute words without worrying too much about loss of information.

Our general algorithm is given in Algorithm 2, and is detailed as follows. For each of the attribute words, we locate each of the instances of such words (line 3). We set a window size of w around the word and run noun chunker[1]. This allows us to selectively run noun chunker on phrases that we are interested in as opposed to running them on entire corpus (line 6 and line 7). We store only the noun phrases that contain the attribute word and disregard the rest. The noun phrases we keep are potential value phrases.

We then rank potential value phrases (line 8). The scoring function is based on summarization techniques that utilize TF-IDF method [31, 33]. TF-IDF approaches balance text frequency (TF) and inverse document frequency (IDF) to extract best set of words that explain multiple documents [31, 33]. We chose the TF-IDF model because numerous other approaches exploit characteristics of how document are written [34] to generate summaries which do not fit our needs. Noun phrases that are extracted as potential values are short, and do not retain the structure of documents. For our approach, we use TF-DF method, where DF is the number of times the words appear across documents to satisfy the second observation we have noted (values on unstructured documents are replicated multiple times). In order to leverage document frequency however, we need to only use words that are of interest when calculating the scoring function. We thus use only nouns, adjectives and numeric values, and prune all stop words for ranking value phrases. Our scoring function for each word is given in Equation 3.5

$$score(w, p, D_i) = tf(w, p) \times df(w, D_i) \tag{3.5}$$

$$tf(w, p) = \frac{fq(w, p) * (k_1 + 1)}{fq(w, p) + k_1} \tag{3.6}$$

$$df(w, D_i) = log\ nq(w, D_i) \tag{3.7}$$

where w is a word in phrase p, fq is number of times w appears in D_i, k_1 is constant similar to those seen in BM25 [35] and nq is number of documents that word w appears in. We

[1]We use noun chunker from www.opennlp.apache.org

then normalize each score to penalize phrases that are too long to derive the following:

$$score(p) = \frac{\sum_{w \in p} score(w, p)}{log(\mid p \mid)} \tag{3.8}$$

Finally, we remove phrases that may not be of interest to us. We have mentioned earlier that there can be multiple values associated with each attributes. It is undesirable to list every values associated with each attributes. Furthermore, some phrases may not even be relevant to values of attributes at all, in which case we remove these phrases.

For each of the scored phrases for each attribute, we first sort them in descending order (line 9). Then, we test for student's t-test on scores of the extracted phrases and remove those with p-value smaller than a pre-set significance (line 10). We set the significance level at 0.05 for all of our tests. Next, we iterate from the highest scoring phrases to lowest scoring phrases, and remove the phrases which consists only of words that have been seen in higher scoring phrases (line 11). As an example, if the two highest scoring values contained words '3.4 Ghz' and 'Intel i7,' then the phrase '3.4 Ghz Intel i7' is removed since it contains all the words that have been seen from higher scoring values. Finally line 12 generates attribute-value pairs (a_{gj}, r) for all remaining ranked values.

Algorithm 2 Value Extraction

Input: Global Attributes A, Documents in topic D_i
Output: Attribute-value tuple (a_{gj}, v_{ijk}) $\forall i, j, k$

```
1: [AVPairs] = φ
2: for all a_gj from A do
3:    [locs] = getAttributeOccurrenceLocations(a_gj, D_i)
4:    [nps] = []
5:    for all location loc in [locs] do
6:       np = chunker(D_i, loc)
7:       append np to [nps]
8:    end for
9:    [rankedValues] = rankValues([nps])
10:   removeLowScoringValues([rankedValues])
11:   removeSubsumedPhrases([rankedValues])
12:   append pairs (a_gj, r), ∀r ∈ rankedValues to [AVPairs]
13: end for
14:
15: return [AVPairs]
```

CHAPTER 4

EVALUATION

Previous works [4, 12, 11, 36] on attributes, or attribute-value pair evaluation relied on using Mechanical Turk [37] or other means of human annotators [38, 39] to evaluate their approaches. While these provide a reliable means of testing the method, it may suffer from lack of standardization of evaluation, or even questionable results because the sample size may be too small.

We instead use a widely used specification page of online electronics store to serve as a gold standard, and introduce new metrics for evaluating attributes, values, and attribute-value pairs. To the best of our knowledge, there are no evaluation techniques that automatically evaluates attribute-value pairs. The closest approaches are in evaluating product sentiments [39, 17] and they not only need human annotators to annotate the correct sentiment of document, but also is a different problem, where they often do not need to evaluate explicit attribute-value pairs.

Leveraging the online store to serve as gold standard has several advantages. First, such specification page has a convenient attribute-value pair that corresponds to the definition we have given in Section 3. Secondly, while the specification page may not list all the corresponding attributes-value pairs that exist for product class, it does cover most, if not all of the pairs of interest. Finally, generating more gold standard is not difficult. Evaluator simply has to write a parser that downloads the specification page for products of interest. There are some limitations however, in that gold standards can only be generated for those that has product page in the online shopping mall. However, for those categories that have products in it, mass generating gold standard is not very difficult.

The approach comes with some limitations as well. Automatic evaluation is unable to detect spelling errors or synonyms of a word potentially lowering the true accuracy of the

method. However, such is likely to be evenly distributed across different approaches and is not a huge limitation. Another limitation lies in that the approach needs some gold standard to compare upon. While we resolve the limitation on our problem by utilizing those that have existing NVP on it, for categories that do not, gold standard still has to be generated.

We have divided our evaluations into three parts. We first describe the dataset and introduce an unsupervised web-query based attribute-value pair extraction method [12] that has been used for comparison and evaluation. Web query approach was chosen because similar to our approach, the method is unsupervised and utilizes unstructured documents. We then evaluate attribute, value and attribute-value pair extractions. For each sections we describe our evaluation criteria. Finally, we show evaluation results from human evaluators.

4.1 Dataset

We generated gold standards from a well known online electronics store[1]. We limited the classes to those that exists in the online store in order to ensure consistency across the standards. There were a total of five classes (laptops, digital cameras, DSLR cameras, tablets and cell phones) and each of the classes has 10 entities. These topics were chosen randomly, with heavier weight towards new products to old ones. We then extracted top 50 results from Bing Search API for each entities to serve as input documents of our algorithm, and then removed all structural cues of each retrieved documents[2]. Unless otherwise noted, all the numbers reported in these sections are based on averaging the scores across the five classes.

In order to compare our method to some of the existing works, we reimplemented an unsupervised, web-query based approach [12]. Their method defines several lexical and syntactical rules to use them in web query. The approach then uses snippets from results of query to obtain attribute-value pairs. We again used Bing API to query the web. In order to extract attributes using web-query based approach, we first attempted to do so by querying each of the entities of a class. However, this turned out to be unfruitful, so we queried

[1] www.newegg.com

[2] We used www.diffbot.com to obtain unstructured data

from the five classes to obtain attributes of entities instead. Values were extracted using the attributes extracted from attribute extraction step. We further note the web query approach had access to more documents than our approach had, because we allowed the method to freely extract texts from entire web.

For the entire evaluation we have preset a few parameters. All the F score were calculated with the same weight given towards both precision and recall, or $\beta = 1$. We have further limited the number of attributes our algorithm extracts. The number was set to 50 for each A_i, and 90 for A_g.

4.2 Attribute Extraction Evaluation

4.2.1 Methodology

Our evaluation methodology for attribute extraction is based on precision and recall. Precision and recall is one of the most widely used evaluation criteria used in information retrieval and forms basis of numerous evaluation approaches. We thus use these two metrics in evaluating our approach. Precision and recall are defined as follows:

$$Precision = \frac{\#retrieved\ and\ relevant}{\#retrieved}$$
$$Recall = \frac{\#retrieved\ and\ relevant}{\#relevant}$$

We would like to evaluate how many of the attributes were detected. The problem with using simple bag-of-words in measuring precision and recall for this problem is that it will inevitably assign higher weights on long attribute words. As an illustrative example, if the gold attribute words are 'memory size' and 'cpu,' and one algorithm has extracted 'memory' and 'size,' and the other method has extracted 'memory,' and 'cpu,' it can easily be seen that the second algorithm had extracted more relevant attribute words than the former. A simple bag-of-words approach scores the output of the two methods with same precision and recall score in this extreme case. We would thus like to measure

1) How many of the gold attributes the algorithm recognized

2) For each of the attributes it has recognized, how close are the retrieved attribute words to that of gold standard.

We address the first question by introducing 'part correct precision/recall' and that of the second one by 'word match precision/recall.' Both of the part correct precision/recall and word match precision/recall are motivated by lexical comparison level [40]. The comparison approach is used in comparing how similar two ontology systems are, and we adopt the idea to extend it to calculating precision and recall.

Lexical comparison level [40] measures the similarity of lexical entries that have been retrieved by two systems. They use Levenshtein's edit distance to define String Matching (SM) between two strings g_i and c_j, and is defined as follows:

$$SM(g_i, c_j) := max(0, \frac{min(|g_i|, |c_j|) - ed(g_i, c_j)}{min(|g_i|, |c_j|)}) \in [0, 1] \tag{4.1}$$

In order to average over all strings $g_i \in G$ and $c_j \in C$, they define averaged String Matching $\overline{SM}(G, C)$:

$$\overline{SM}(G, T) := \frac{1}{G} \sum_{g_i \in G} max_{c_j \in C} SM(g_i, c_j) \tag{4.2}$$

It should be noted that Equation 4.1 measures how many strings are matched between g_i and c_j, or $match(g_i, c_j) = min(|g_i|, |c_j|) - ed(g_i, c_j)$. Furthermore, in order to make denominator that of a precision or recall, instead of taking $min(|g_i|, |c_j|)$, we divide them by the length of string of precision or recall.

Our word match correct precision/recall is thus derived directly from Equation 4.2, with the exception of denominators which are normalized according to the definition of precision and recall. The following is the equation :

$$Precision_{wm} = \frac{1}{\mid C \mid} \sum_{c \in C} max_{g \in G} \frac{match(g, c)}{\mid c \mid}$$

$$Recall_{wm} = \frac{1}{\mid G \mid} \sum_{g \in G} max_{c \in C} \frac{match(g, c)}{\mid g \mid}$$

where $match(g, c)$ measures the number of words that match each other. Such methodology allows us to measure how many of attributes are matched. Furthermore, it allows us to see if the word is fully matched or partially matched. It is, however, often useful to know how many attributes have been noted by the algorithm eventhough the words may not fully match the given gold standard. We thus introduce part correct precision and recall metric, and they are given as follows :

$$Precision_{pc} = \frac{1}{\mid C \mid} \sum_{c \in C} max_{g \in G}\ match_{cell}(g, c)$$

$$Recall_{pc} = \frac{1}{\mid G \mid} \sum_{g \in G} max_{c \in C}\ match_{cell}(g, c)$$

where $match_{cell}(g, c)$ returns 1 if at least one of the words in g and c match, and 0 otherwise.

4.2.2 Performance Comparison

In this section, we would like to answer

1) Would adding more entities help in extracting attributes, and if so, by how much?

2) What are the impact of adding cross entropy terms, and using only global attribute extractor A_g or intersection of local attribute extractors A_i?

3) How well does our approach work compared to web query approach [12] in terms of part precision/recall and word match precision/recall?

In order to evaluate performance gains by adding more entity of a topic, we ran each rounds varying the number of entities used for class attribute extraction. Each of the rounds were averaged on different subset of entities to ensure consistencies across the class. Such redundancy ensures no one particular run performs better than the other by differences in dataset.

It can be seen from Figure 4.1 that adding more entity improves both $F_{part\ correct}$ and $F_{word\ match}$ of attribute extraction. In particular, F scores increase logarithmically. Noteworthy performance gain is seen with relatively few entities in a class. Adding more entities, as is expected, leads to more redundancies in attributes extracted, which our approach uti-

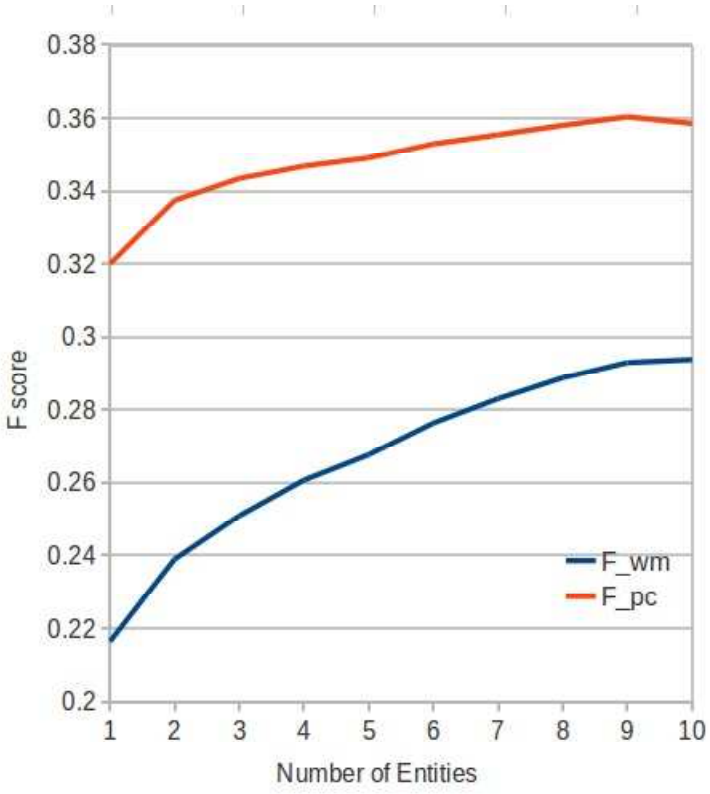

Figure 4.1: F score as number of entities increases.

lizes in extracting attributes. However, as more entities are added in extracting attributes, most of the redundant attributes have been noted, and the F scores do not increase as much.

For our second experiment, we compare how adding cross document entropy term helps. We have also argued having global attribute extractor A_g and local attribute extractor A_i improves performance over having single A_g or taking intersection with all local attribute extractor ($A = A_i \cap A_j \ \forall i, j$) to obtain attribute set A. We show the effects of combining A_g and A_i over other approaches as well.

Our experimental results comparing performance for these different types of attribute extraction is shown in Figure 4.2. Having entropy term has helped both the recall and precision on all the criteria, verifying our hypothesis that attributes tend to appear often across the documents. Intersection approach outperforms all the other algorithm in terms of precision, at the expense of recall. This is expected because attribute is added only if it appears across all entities of the class, forcing only the very high quality attributes to be added. This, in turn, comes at the expense of recall. Global approach is somewhat different,

where it tends to perform better on recall at the cost of precision. The approach relies on only one attribute extractor, A_g as opposed to exploiting redundancy. The redundancy improves precision, removing potential noise attributes.

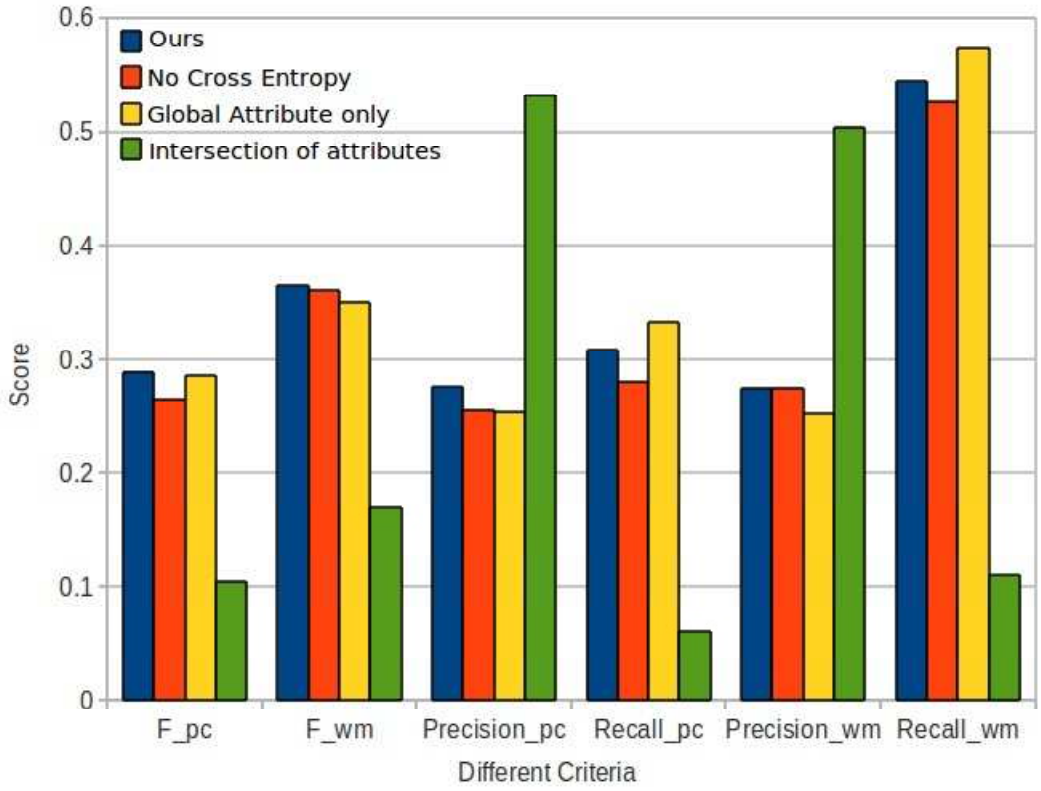

Figure 4.2: Blue : Our approach, red : no cross entropy term, yellow : using only global extractor A_g and green : intersection of all local attributes

Table 4.1 shows the performance comparison of the web query and our approach. The table reports average precision, recall and F score of both of the approaches. We see that the web query approach [12] has slightly better precision while our approach is more of a recall based. Our approach performs better overall, with a higher F-score than the web-query based one. Finally, Table 4.2 shows attributes that have been extracted by the algorithm.

4.3 Value Extraction Evaluation

Our value evaluation is based on ROUGE evaluation [10]. ROUGE is a recall-based evaluation technique that is widely used in automated text summarization. It has been shown that

Scoring Criteria	Our approach	Web Query
F_{pc}	**0.36**	0.25
F_{wm}	**0.29**	0.17
$Precision_{pc}$	0.27	**0.33**
$Recall_{pc}$	**0.54**	0.22
$Precision_{wm}$	0.27	**0.35**
$Recall_{wm}$	**0.31**	0.12

Table 4.1: Comparison of attribute extraction performance

the scores that they assign correlate with those of human judgment. We adapt ROUGE-1 which is based on unigram match to evaluate extracted values. Unlike text summarization tasks, value phrases do not need to match in exact order at which it appears. As an illustrative example, consider a phrase 'Intel Core i5 2.5Ghz.' Even if one of the algorithm extracted 'Intel 2.5Ghz core i5,' the underlying concept is the same. Furthermore, as both of the algorithms are extractive methods, it is unlikely that they would have extracted phrases with vastly different word ordering from gold standard.

We further note that there may be multiple (a_{ij}, v_{ijk}) pairs for some attribute a_{ij}, while for other attributes a'_{ij} there may be only one value associated with it. Because of this characteristics, we introduce two different approaches in evaluating extracted values. The first one corresponds to 'global,' which corresponds to averaging scores of all the values that has been extracted across all attributes. 'Local' score calculates ROUGE-1 scores for all the values associated with each of the attributes. It then takes the average of the calculated ROUGE-1 scores across attributes.

More formally, we define global score as

$$\frac{1}{\mid V_i \mid} \sum_{v \in V_i} ROUGEscore(v)$$

where V_i is a set of all values assigned to topic i and local score as

$$\frac{1}{\mid A_g \mid} \sum_{a \in a_g} \frac{1}{\mid values(a) \mid} \sum_{v \in values(a)} ROUGEscore(v)$$

As an example, let us say there are three and five values associated with attributes CPU

Class	Attributes
Laptops	hd, gb, laptop, notebook, processor, usb, intel, core, battery, 4gb, pc, web, amd, internet, online, bluetooth, product, graphic, adapter, premium, click, aluminum, shipping, user, video, vga, store, feature, display, specification, integrate, audio, keyboard, backlit, port, ram, memory, dai, wireless, warranty, color, connectivity, card, connect, fingerprint, technology, performance, ssd, price, digital, graphics, sleek, ethernet, spec, mhz, networking, headphone, gaming, email, wa, design, thunderbolt, storage, device, computing, built-in, 8gb, os, powerful, drive, model, purchase
Digital Cameras	zoom, lcd, camera, iso, auto, shutter, color, digital, sensor, optical, lens, mode, image, shooting, photo, feature, compact, hd, capture, stabilization, shoot, video, resolution, usb, battery, panorama, detection, af, ccd, flash, macro, in-camera, button, shot, wide-angle, focus, tripod, playback, screen, waterproof, pixel, photography, exposure, movie, portrait, tracking, underwater, picture, sensitivity, aa, smart, blur, menu, viewfinder, panoramic, select, telephoto, card, specification, scene, built-in, fp, recording, frame, monochrome, filter, high-resolution, charger, memory, autofocus
DSLR Camers	lcd, sensor, hd, iso, color, lens, af, shutter, camera, viewfinder, auto, shooting, autofocus, jpeg, image, digital, mode, slr, flash, aperture, photo, fp, built-in, button, movie, feature, cmos, zoom, video, manual, battery, compact, nikon, capture, shoot, exposure, usb, interchangeable, in-camera, metering, pixel, continuous, micro, playback, improv, resolution, frame, photographer, dial, focus, panasonic, screen, speed, kit, accessory, filter, shot, noise, wa, raw, photography, tracking, compatible, 3d, quality, menu, macro, recording, format, microphone, option, adjustment, user, brightness, picture, optional, stereo, detection, mount, capturing
Tablets	android, processor, gb, bluetooth, screen, tablet, hd, honeycomb, device, battery, web, app, camera, usb, video, slot, built-in, pc, color, mobile, display, sd, capacitive, browsing, flash, keyboard, connectivity, feature, port, gaming, photo, favorite, dai, internet, ips, wireless, multitasking, sync, design, micro, download, aluminum, user, interface, laptop, adobe, email, touch, graphic, online, operating, 1gb, 8gb, access, docking, stereo, apps, functionality, storage, ios, game, enjoy, slate, os, fingerprint, dual, ram, card, rear-facing, resolution, store, specification, amazon, memory, optimize, browse, 3d, button, content, headphone
Cellphones	bluetooth, color, camera, mobile, phone, android, screen, battery, device, email, app, gsm, feature, download, usb, handset, browser, gb, internet, messaging, mah, video, button, mb, photo, update, qwerty, pixel, web, dai, headphone, sensor, keyboard, gps, os, mhz, wireless, flash, twitter, processor, samsung, user, wa, hd, unlock, display, favorite, headset, connectivity, browsing, capacitive, gingerbread, network, ios, resolution, 4g, card, audio, charger, navigation, widget, networking, store, tri, sync, touch, interface, built-in, droid, online, sms, 4s, version, 2g, menu, slot

Table 4.2: Extracted attributes

and memory respectively. For simplicity, let us assume all the values have ROUGE score of σ_c for CPU, and σ_m for memory. Local score would then be $\frac{\frac{3\sigma_c}{3}+\frac{5\sigma_m}{5}}{2}$ since there are two attributes, while global score is $\frac{3\sigma_c+5\sigma_m}{8}$, since there are eight values in total.

For value evaluation, we are interested answering two questions. The first one concerns design choices that were made in value extraction. We compare between the proposed approach, length normalization, and removing stop words. Adding length normalization and removing stop words both helped in improving both local and global value scores. The second evaluation compares against web query. Our approach performed better than that of web query approach. We suspect because web query approach is based on pattern-matching, their algorithm was unable to find sufficient number of web documents on specific entity. Both the design choice evaluation and value extraction performance comparison are shown in Figure 4.3.

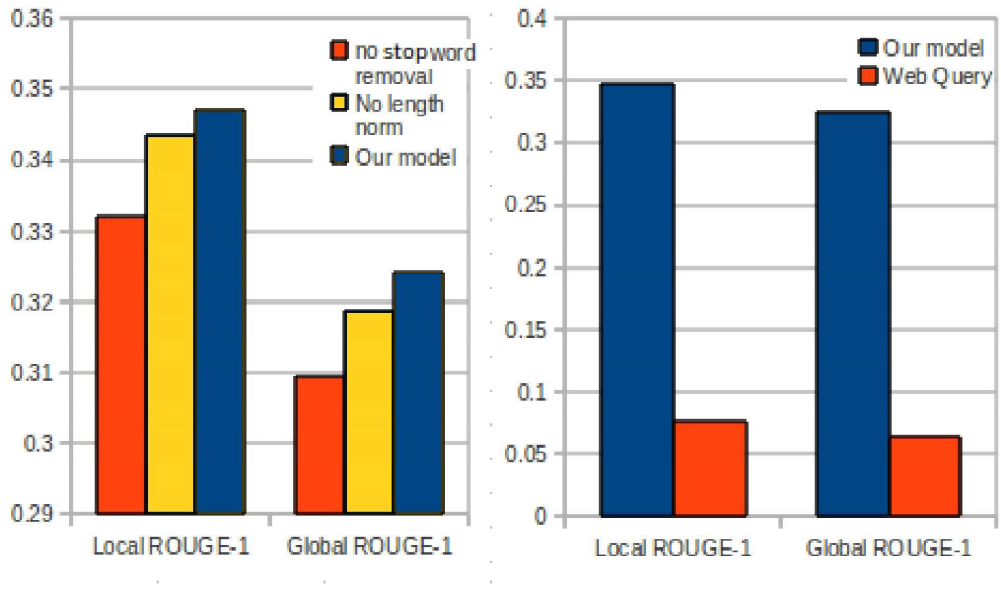

Figure 4.3: Left : Design choice evaluation, Right : Comparison between our approach and that of web query

attributes	Acer Aspire S3-951-6828	Apple MacBook Pro MD314LL	Gateway NV55S15u	ASUS Zenbook UX31E-DH52
drive	gb 5,400 rpm hard drive	gb serial ata hard drive (5400 rpm) 4 gb install ram	1.5 tb sata hard drive (green product	sata revision 3.0 solid state drive and usb
port	the hdmi port	gigabit ethernet port, usb 2.0 port	usb 2.0 port hdmi port, video port	usb 3.0 , mini display port
processor	intel core i5-2467m processor (1.6 ghz	intel core i7 processor 4 gb 1333 mhz, intel core 2 duo processor	amd quad-core a8-3500m accelerated processor 1.5 ghz 17.3 " hd	intel core i5 processor
ram	4gb ram, ddr3 ram	4gb ram	gateway lt2016u netbook 1gb ram	4gb ddr3 ram
battery	connectivity and long battery life	notebook lithium polymer battery	6-cell li-ion battery (4400 mah)	6-7 hour battery life rate

Table 4.3: Example attribute-value output from the program

4.4 Attribute-value pair Comparison

We introduce attribute-value pairs evaluation in this section. While it may seem that merely evaluating attributes and values may be enough to judge the system, it is possible that attribute and value both corresponds to some entry in gold standard, but they may not be aligned and not form a valid (a_{ij}, v_{ijk}) tuple. As an example, an approach may have extracted 'CPU' as attribute and '8GB' as value. While both the attribute and value are valid characteristics of laptop, the pair do not align. We introduce attribute-value score to calculate alignment by utilizing the two criteria we have introduced in the previous subsections.

Similar to the strategy used in attribute extraction evaluation, attribute-value pair score is given by

$$Precision = \frac{1}{|A_T|} \sum_{(a_t, v_t) \in A_T} score((a_t, v_t), A_G)$$

$$Recall = \frac{1}{|A_G|} \sum_{(a_g, v_g) \in A_G} score((a_g, v_g), A_T)$$

$$score((a, v), A) = max_{(a', v') \in A} sim((a, v), (a', v'))$$

where A_T refers to sets of all attribute-value pair (a_t, v_t) from candidate set and A_G refers

Scoring Criteria	Ours	Web Query
F_{pc},ROUGE-1	**0.19**	0.11
F_{wm},ROUGE-1	**0.17**	0.09

Table 4.4: Attribute-value pair performance.

to that of gold set. The similarity function is given by

$$sim((a, v), (a', v')) = \alpha \cdot score(a, a') + (1 - \alpha) \cdot score(v, v')$$

where α denotes how much weight to give to attribute. Bigger α leads to assigning more weights to correctly labeled attributes and vice versa. For all our experiments we set $\alpha = 0.5$, giving equal weight to both attributes and values. The scoring functions, $score(a, a')$ and $score(v, v')$ can be any of the attribute or value extraction criteria we have defined in the previous sections. We use F_{pc} and F_{wm} for attribute score, and ROUGE-1 for value score. In Table 4.4 we compare attribute-value score of our approach with that of web query approach. The qualitative results are shown in Table 4.3. It can be seen that many relevant values are extracted with some noise (such as 17.3 " hd for the attribute processor). Noises arise from the limitation of chunker as it was unable to correctly trim out the relevant noun phrases.

4.5 Human Evaluation

We also measure human evaluation to gauge how well the algorithm performed against the web-query based approach. We asked eight human evaluators of various backgrounds (from College of Business, Liberal Arts, and Engineering) who were either undergraduate or graduate students to partake in our evaluation. Undergraduate and graduate students are among the most technology-savy group, and choosing from this subgroup allowed us to avoid questionable annotations. At the same time, having various departmental background allowed generality across different types of users, and that the utility of the method isn't limited to subset of users, particularly to those studying computers. Furthermore, in order to ensure the users were clear about the task at hand, we made sure all the evaluators were native speakers of English.

Evaluators were then split into two groups. The first group measured the performance of recall for both algorithms. Results from our approach and web query algorithm were placed into two different files. We refer to these files as file 'T' and file 'W' for brevity purposes. The evaluators were then presented with attribute and value pair from the online shopping mall, which again acted as a gold standard. For each items in gold standard, users had to answer three questions for two files, for a total of six questions. These questions were 'Does this attribute appear in file T/W?,' 'Does this value appear in file T/W?' and 'Does this attribute-value pair appear in file T/W?' The evaluators were not informed of which file corresponded to which algorithm.

The second group were tested for precision of attribute-value extraction. Attribute-value pair results from our algorithm and web query based algorithm were shuffled and presented to evaluators. Evaluators were then asked to determine if attribute was valid or not. Similar question was asked for value evaluation. Finally, evaluators were asked if attribute-value pairs aligned well. There may have been a case where attribute and value were both correct, but they may not align. Such examples include 'CPU' and '8GB,' where both of them pertains to valid attribute and value, but do not align. Our results can be seen in Table 4.5.

From both the automatic and manual evaluations, we conclude that 1) leveraging multiple entities help in improving attribute extraction performance and 2) our approach is better suited on value extraction than web query approach, especially if topics become more specific. Web query approach relies on matching patterns, and as topics become more specific, finding predefined pattern becomes harder. There are less documents talking about specific topic than a broad one. We rely on ranking potential value phrases based on text summarization technique, avoiding potential sparsity issues that may be brought up by pattern matching based methods.

Attribute			
Approach	Precision	Recall	F-Score
Our Approach	0.85	**0.34**	**0.48**
Web Query	**0.90**	0.25	0.40
Value			
Approach	Precision	Recall	F-Score
Our Approach	**0.89**	**0.46**	**0.60**
Web Query	0.79	0.22	0.34
NVP			
Approach	Precision	Recall	F-Score
Our Approach	**0.69**	**0.17**	**0.28**
Web Query	0.57	0.14	0.22

Table 4.5: Human evaluation results on class 'laptop.'

CHAPTER 5

CONCLUSION AND FUTURE WORKS

We have introduced an unsupervised approach to extracting attribute-value pairs on unstructured documents, and demonstrated that the performance of unsupervised attributed extraction can be significantly improved by combining multiple entities of the same class. The approach is general in that it can be run on not only the product but also on reviews. Furthermore, our experiments show that the performance of attribute extraction is logarithmic in the number of available entities, and it is therefore sufficient to obtain only a few number of entities (between 5-10) of the same class for significant performance gains. It is possible to use the idea of combining entities of the same topic to enhance the performance of other attribute extraction algorithms. Next, we extended an existing summarization technique to extract values. Such technique was able to extract meaningful value phrases. As we have hypothesized earlier on, pattern-based approaches or those that are based on such appraoches do not perform very well on web-scale dataset.

We further proposed an automatic attribute-value pair evaluation based on Lexical comparison level [40] and ROUGE evaluation [10]. Many of the previous approaches rely on human evaluators to judge their algorithm. However, because there is no standardized dataset, it is not easy to compare different attribute-value pair extraction methods. We propose to generate a gold-standard dataset by exploiting a widely used online shopping mall, noting that while this may not cover all the attributes-values of interest, it covers vast majority of those that the users may be interested in.

For future works, more works can be investigated in value extraction to combine a more sophisticated summarization techniques, such as those motivated by LDA-based approaches, lexical analysis, or coreference analysis may help in enhancing the quality of value extractions. Furthermore, we did not remove irrelevant documents for our data. It is possible to

utilize background documents [41] to remove irrelevant articles to improve the performance. It would be of interest to develop approaches that can quantitatively evaluate attribute-value extraction systems on entities for which such data are not available (e.g. medical forums or biography), using the same terminology we have used to evaluate products. Unlike producing gold standard for products however, there are no easily extractable structured gold standard which may make evaluation more challenging to be fair and quantifiable.

REFERENCES

[1] B. Chee, R. Berlin, and B. Schatz, "Measuring population health using personal health messages," ser. AMIA Annual Symposium, 2009.

[2] C. Kwok, O. Etzioni, and D. S. Weld, "Scaling question answering to the web," *ACM Trans. Inf. Syst.*, vol. 19, no. 3, pp. 242–262, July 2001. [Online]. Available: http://doi.acm.org/10.1145/502115.502117

[3] M. J. Cafarella and E. Wu, "Uncovering the relational web," in *In under review*, 2008.

[4] A. Kopliku, M. Boughanem, and K. Pinel-Sauvagnat, "Towards a framework for attribute retrieval." in *CIKM*, C. Macdonald, I. Ounis, and I. Ruthven, Eds. ACM, 2011, pp. 515–524.

[5] M. J. Cafarella, E. Wu, A. Halevy, Y. Zhang, and D. Z. Wang, "Webtables: Exploring the power of tables on the web," 2008.

[6] H.-H. Chen, S. chung Tsai, and J.-H. Tsai, "Mining tables from large scale html texts," 2000, pp. 166–172.

[7] K. Probst, R. Ghani, M. Krema, A. Fano, and Y. Liu, "Semi-supervised learning of attribute-value pairs from product descriptions," in *In IJCAI-07, Ferbruary*, 2007.

[8] R. Girju, A. Badulescu, and D. I. Moldovan, "Learning semantic constraints for the automatic discovery of part-whole relations," in *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*. Morristown, NJ, USA: Association for Computational Linguistics, May 2003, pp. 1–8.

[9] S. Soderland, "Learning to extract text-based information from the world wide web," *Proceedings of Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, 1997.

[10] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Proc. ACL workshop on Text Summarization Branches Out*, 2004. [Online]. Available: http://research.microsoft.com/ cyl/download/papers/WAS2004.pdf p. 10.

[11] C. Scaffidi, K. Bierhoff, E. Chang, M. Felker, H. Ng, and C. Jin, "Red opal: product-feature scoring from reviews," in *ACM Conference on Electronic Commerce*, 2007, pp. 182–191.

[12] D. Sánchez, "A methodology to learn ontological attributes from the web," *Data Knowl. Eng.*, vol. 69, pp. 573–597, June 2010. [Online]. Available: http://dx.doi.org/10.1016/j.datak.2010.01.006

[13] M. Hu and B. Liu, "Mining opinion features in customer reviews," in *Proceedings of AAAI*, 2004, pp. 755–760.

[14] A.-M. Popescu and O. Etzioni, "Extracting product features and opinions from reviews," in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, ser. HLT '05. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005. [Online]. Available: http://dx.doi.org/10.3115/1220575.1220618 pp. 339–346.

[15] P. D. Turney, "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002. [Online]. Available: http://dx.doi.org/10.3115/1073083.1073153 pp. 417–424.

[16] E. Hovy and C.-Y. Lin, "Automated text summarization and the summarist system," in *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*, ser. TIPSTER '98. Stroudsburg, PA, USA: Association for Computational Linguistics, 1998. [Online]. Available: http://dx.doi.org/10.3115/1119089.1119121 pp. 197–214.

[17] H. D. Kim and C. Zhai, "Generating comparative summaries of contradictory opinions in text," in *Proceedings of the 18th ACM conference on Information and knowledge management*, ser. CIKM '09. New York, NY, USA: ACM, 2009. [Online]. Available: http://doi.acm.org/10.1145/1645953.1646004 pp. 385–394.

[18] M. J. Paul, C. Zhai, and R. Girju, "Summarizing contrastive viewpoints in opinionated text," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1870658.1870665 pp. 66–76.

[19] J. Pustejovsky, "The generative lexicon," *Comput. Linguist.*, vol. 17, no. 4, pp. 409–441, Dec. 1991. [Online]. Available: http://dl.acm.org/citation.cfm?id=176321.176324

[20] K. Tokunaga and K. Torisawa, "Automatic discovery of attribute words from web documents," in *In Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP-05), pages 106118, Jeju Island, Korea*, 2005, pp. 106–118.

[21] I. Titov and R. McDonald, "Modeling online reviews with multi-grain topic models," 2008.

[22] Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai, "Topic sentiment mixture: modeling facets and opinions in weblogs," in *In Proc. of the 16th Int. Conference on World Wide Web*, 2007, pp. 171–180.

[23] S. Brody and N. Elhadad, "An unsupervised aspect-sentiment model for online reviews," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, ser. HLT '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1857999.1858121 pp. 804–812.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944937

[25] N. Yoshinaga and K. Torisawa, "Open-domain attribute-value acquisition from semi-structured texts," in *Proceedings of the workshop on Ontolex*, 2007, pp. 55–66.

[26] B. Wu, X. Cheng, Y. Wang, Y. Guo, and L. Song, "Simultaneous product attribute name and value extraction from web pages," in *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, ser. WI-IAT '09. Washington, DC, USA: IEEE Computer Society, 2009. [Online]. Available: http://dx.doi.org/10.1109/WI-IAT.2009.286 pp. 295–298.

[27] V. Crescenzi and G. Mecca, "Automatic information extraction from large websites," *J. ACM*, vol. 51, no. 5, pp. 731–779, Sep. 2004. [Online]. Available: http://doi.acm.org/10.1145/1017460.1017462

[28] I. Muslea, S. Minton, and C. Knoblock, "A hierarchical approach to wrapper induction," in *Proceedings of the third annual conference on Autonomous Agents*, ser. AGENTS '99. New York, NY, USA: ACM, 1999. [Online]. Available: http://doi.acm.org/10.1145/301136.301191 pp. 190–197.

[29] F. Wu and D. S. Weld, "Autonomously semantifying wikipedia," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, ser. CIKM '07. New York, NY, USA: ACM, 2007. [Online]. Available: http://doi.acm.org/10.1145/1321440.1321449 pp. 41–50.

[30] B. Liu, C. W. Chin, and H. T. Ng, "Mining topic-specific concepts and definitions on the web," in *Proceedings of the 12th international conference on World Wide Web*, ser. WWW '03. New York, NY, USA: ACM, 2003. [Online]. Available: http://doi.acm.org/10.1145/775152.775188 pp. 251–260.

[31] K. R. McKeown, J. L. Klavans, V. Hatzivassiloglou, R. Barzilay, and E. Eskin, "Towards multidocument summarization by reformulation: progress and prospects," in *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, ser. AAAI '99/IAAI '99. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1999. [Online]. Available: http://dl.acm.org/citation.cfm?id=315149.315355 pp. 453–460.

[32] D. R. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization," *Comput. Linguist.*, vol. 28, no. 4, pp. 399–408, Dec. 2002. [Online]. Available: http://dx.doi.org/10.1162/089120102762671927

[33] D. Radev, H. Jing, M. Sty, and D. Tam, "Centroid-based summarization of multiple documents," 2004.

[34] D. Das and A. F. T. Martins, "A survey on automatic text summarization," 2007.

[35] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[36] S. Raju, P. Pingali, and V. Varma, "An unsupervised approach to product attribute extraction," in *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ser. ECIR '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 796–800.

[37] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing user studies with mechanical turk," in *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, ser. CHI '08. New York, NY, USA: ACM, 2008. [Online]. Available: http://doi.acm.org/10.1145/1357054.1357127 pp. 453–456.

[38] K. Ganesan and C. Zhai, "Opinion-based entity ranking," *Information Retrieval*, 2011.

[39] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '04. New York, NY, USA: ACM, 2004. [Online]. Available: http://doi.acm.org/10.1145/1014052.1014073 pp. 168–177.

[40] A. Maedche and S. Staab, "Measuring similarity between ontologies," in *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, ser. EKAW '02. London, UK, UK: Springer-Verlag, 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=645362.650859 pp. 251–263.

[41] M. Efron, P. Organisciak, and K. Fenlon, "Building topic models in a federated digital library through selective document exclusion," 2010.